

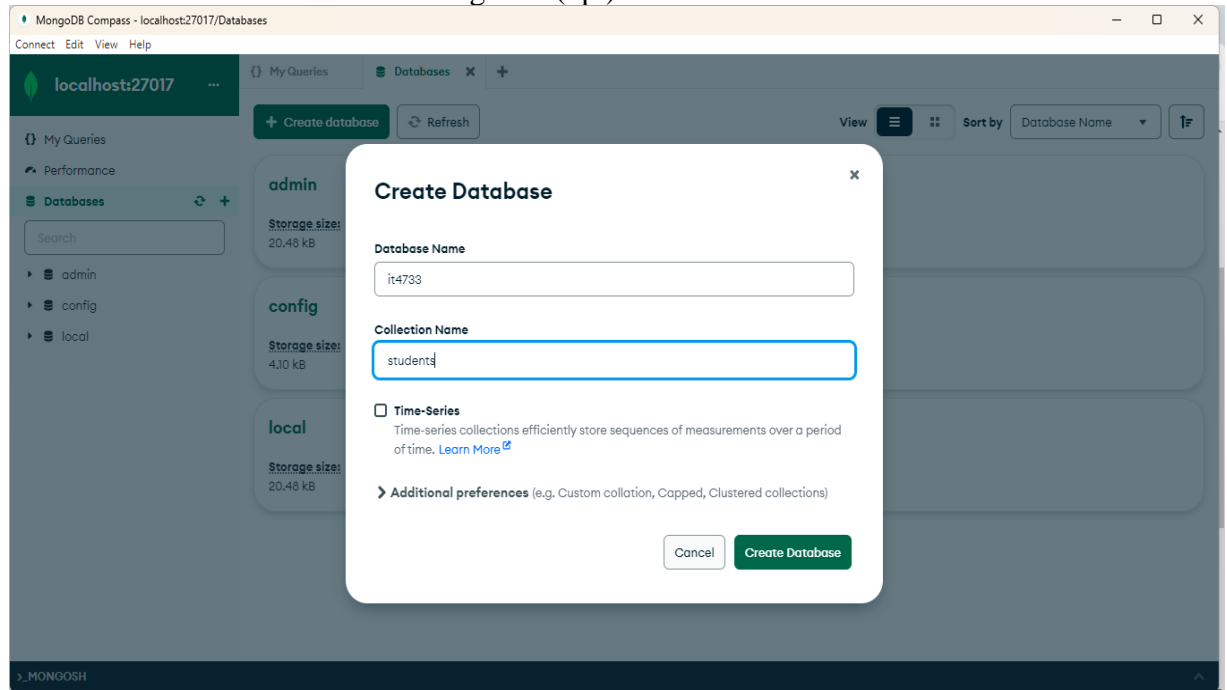
# Project 2

## Objectives

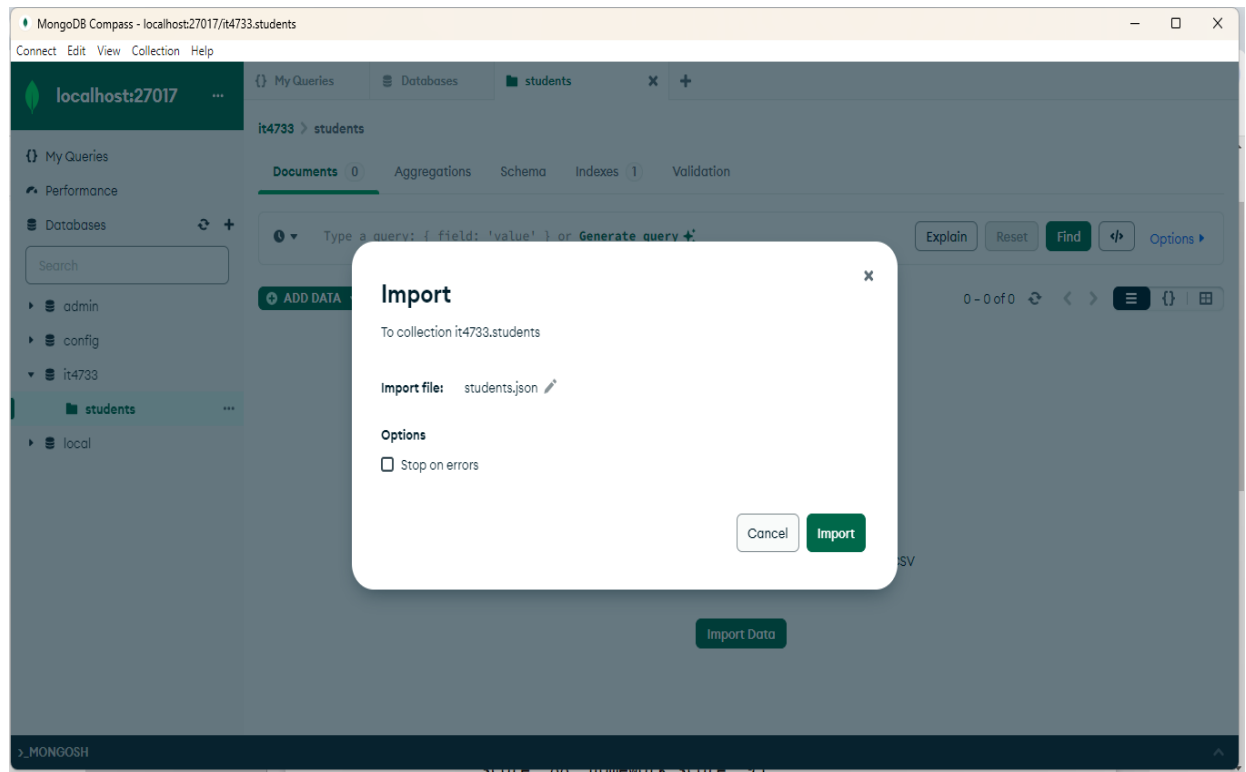
- Practice with MongoDB. This project is more structured than project 1 in that you will import a database to MongoDB and work on the following tasks. For all task, please include the MongoDB code to perform, as well as a screenshot of the result. Please include all codes and screenshots in one word/pdf document and submit for your project.

## Tasks

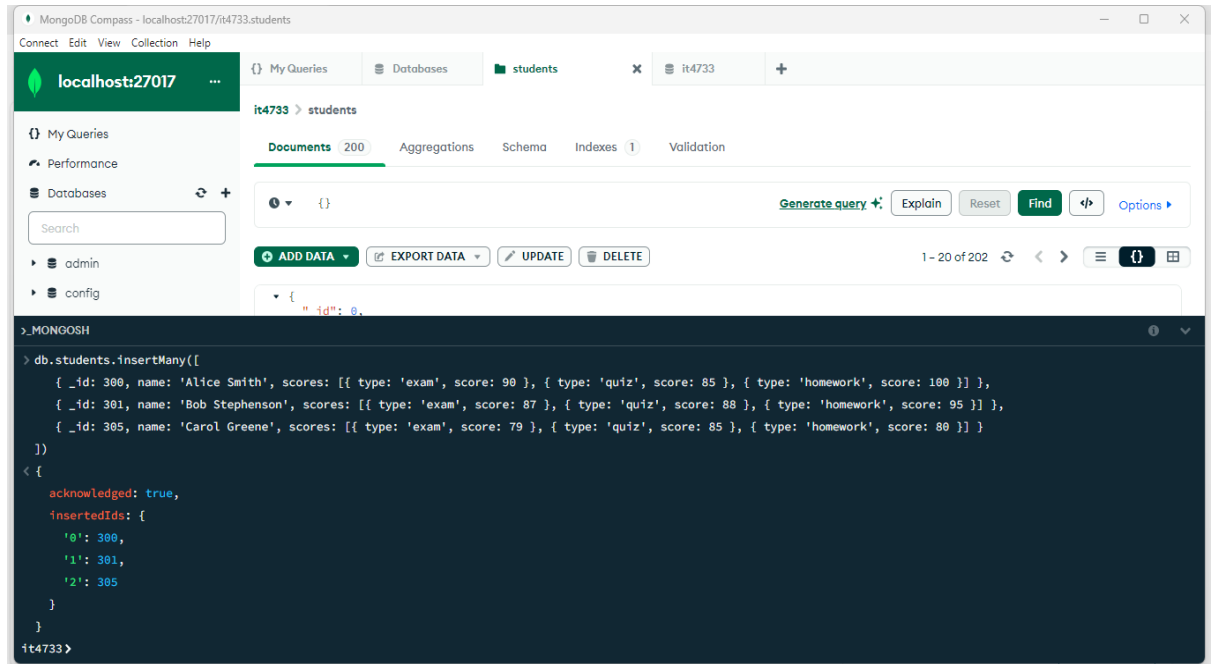
1. Create an “**it4733**” database in MongoDB (4pt)



2. Download the *students.json* data from <https://github.com/ozlerhakan/mongodb-json-files/blob/master/datasets/students.json>. Then, import the data into the **it4733** database, collection **students**. (8pt)

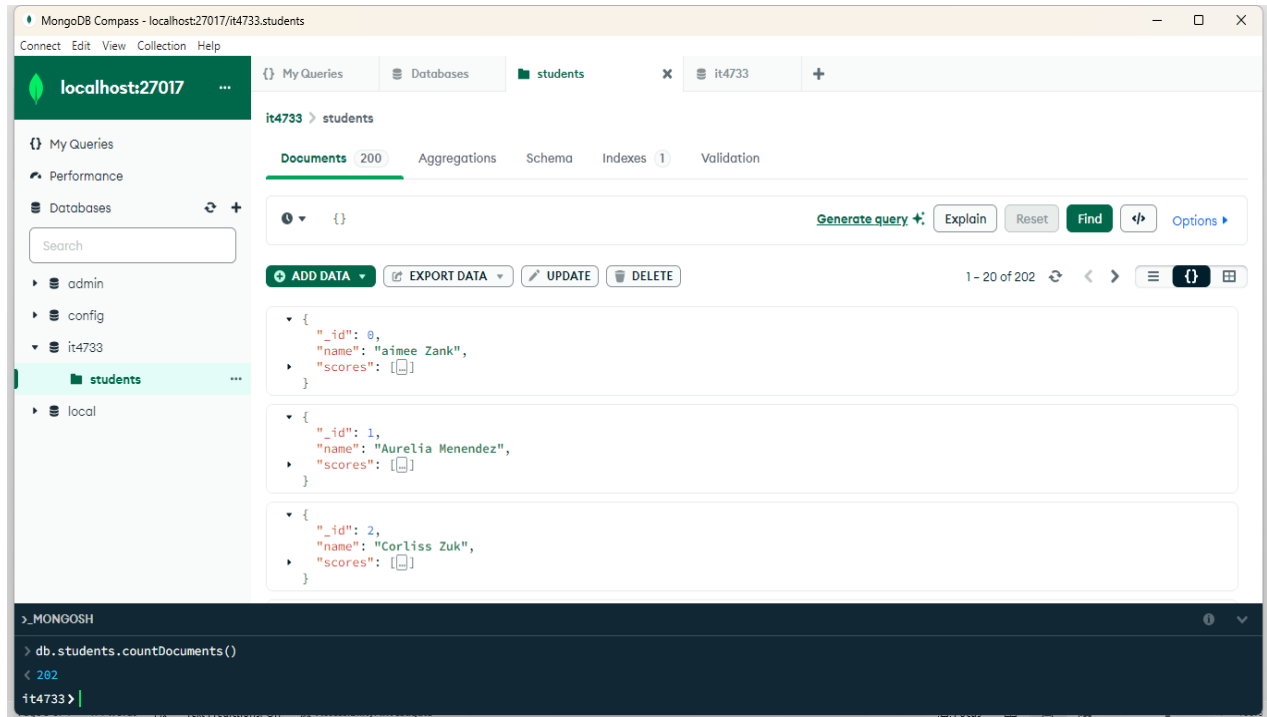


3. Write and run a MongoDB command to insert the following three students into the collection (8pt)
1. `_id: 300`, name: Alice Smith, exam score: 90, quiz score:85, homework score: 100
  2. `_id: 301`, name: Bob Stephenson, exam score: 87, quiz score: 88, homework score: 95
  3. `_id: 305`, name: Carol Greene, exam score: 79, quiz score:85, homework score: 80



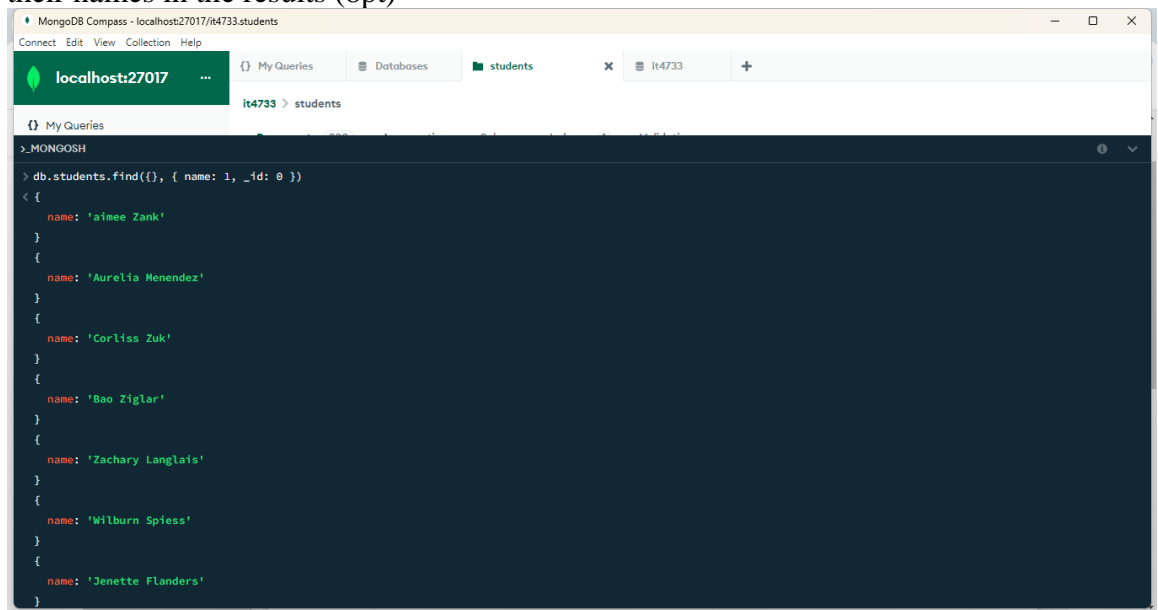
**Code:** `db.students.insertMany([  
 { _id: 300, name: 'Alice Smith', scores: [{ type: 'exam', score: 90 }, { type: 'quiz', score: 85 }, { type: 'homework', score: 100 } ] },  
 { _id: 301, name: 'Bob Stephenson', scores: [{ type: 'exam', score: 87 }, { type: 'quiz', score: 88 }, { type: 'homework', score: 95 } ] },  
 { _id: 305, name: 'Carol Greene', scores: [{ type: 'exam', score: 79 }, { type: 'quiz', score: 85 }, { type: 'homework', score: 80 } ] }  
])`

4. Write and run a MongoDB command to count the number of students in the collection (8pt)



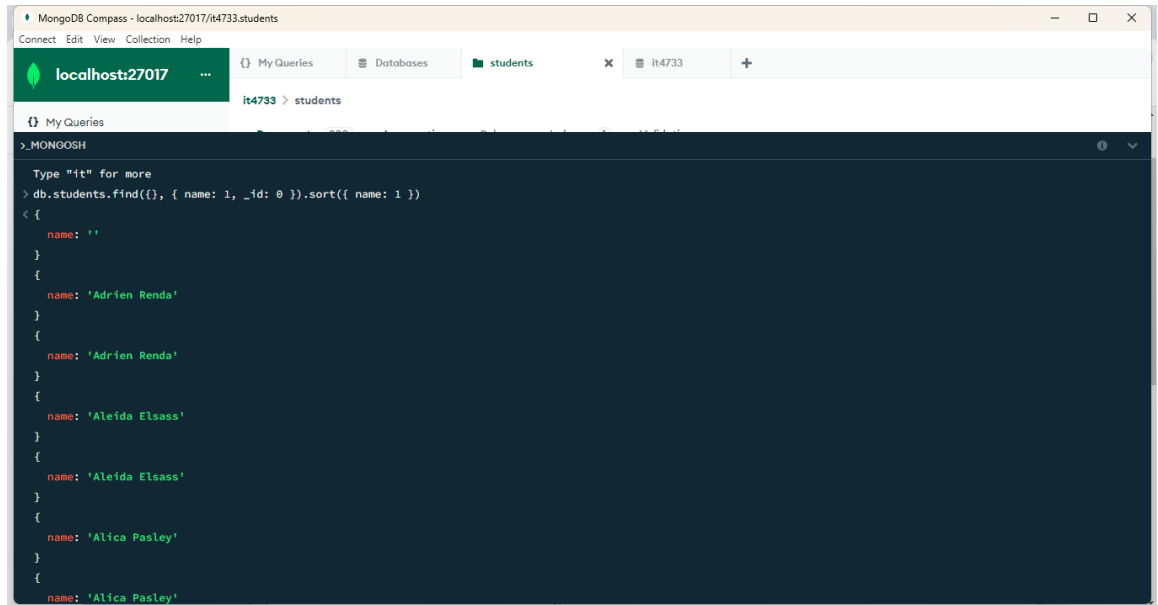
**Code: db.students.countDocuments()**

5. Write and run a MongoDB command to query all the students and only display their names in the results (8pt)



**Code: db.students.find({}, { name: 1, \_id: 0 })**

6. Write and run a MongoDB command to query all the students and only display their names in the results, however, sort the results by names (8pt)

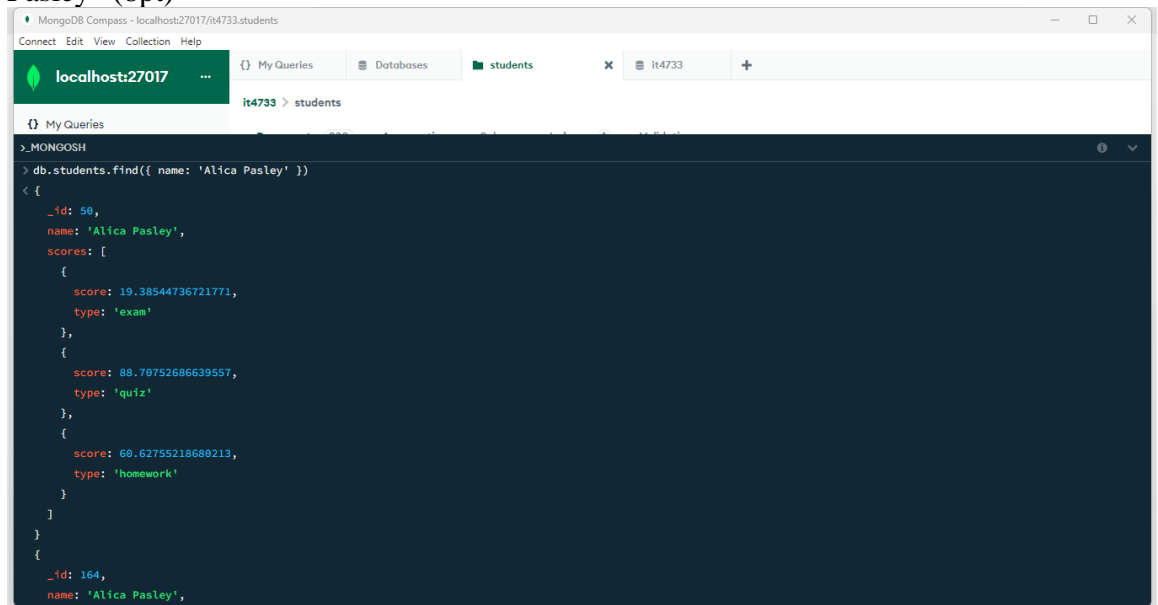


```
localhost:27017 ...
My Queries
Databases
students
it4733 +

My Queries
MONGOSH
Type "it" for more
> db.students.find({}, { name: 1, _id: 0 }).sort({ name: 1 })
< {
  name: ''
}
{
  name: 'Adrien Renda'
}
{
  name: 'Adrien Renda'
}
{
  name: 'Aleida Elsass'
}
{
  name: 'Aleida Elsass'
}
{
  name: 'Alica Pasley'
}
{
  name: 'Alica Pasley'
}
```

**Code:** `db.students.find({}, { name: 1, _id: 0 }).sort({ name: 1 })`

- Write and run a MongoDB command to query all students whose name is “Alica Pasley” (8pt)



```
localhost:27017 ...
My Queries
Databases
students
it4733 +

My Queries
MONGOSH
> db.students.find({ name: 'Alica Pasley' })
< {
  _id: 50,
  name: 'Alica Pasley',
  scores: [
    {
      score: 19.38544736721771,
      type: 'exam'
    },
    {
      score: 88.78752686639557,
      type: 'quiz'
    },
    {
      score: 60.62755218680213,
      type: 'homework'
    }
  ]
}
{
  _id: 164,
  name: 'Alica Pasley',
  scores: [
    {
      score: 19.38544736721771,
      type: 'exam'
    },
    {
      score: 88.78752686639557,
      type: 'quiz'
    },
    {
      score: 60.62755218680213,
      type: 'homework'
    }
  ]
}
```

**Code:** `db.students.find({ name: 'Alica Pasley' })`

- Write and run a MongoDB command which is equivalent to "SELECT \* FROM students WHERE name LIKE '%ski'" (8pt)

The screenshot shows the MongoDB Compass interface. The top bar indicates the connection to 'localhost:27017' and the database 'it4733'. The 'students' collection is selected. The query editor shows the command: `db.students.find({ name: { $regex: 'ski$', $options: 'i' } })`. The results pane displays two documents: one for 'Verdell Sowinski' with scores of 62.12870233199035 (exam), 84.74586220889356 (quiz), and 81.58947824932574 (homework); and another for 'Ta Sikorski'.

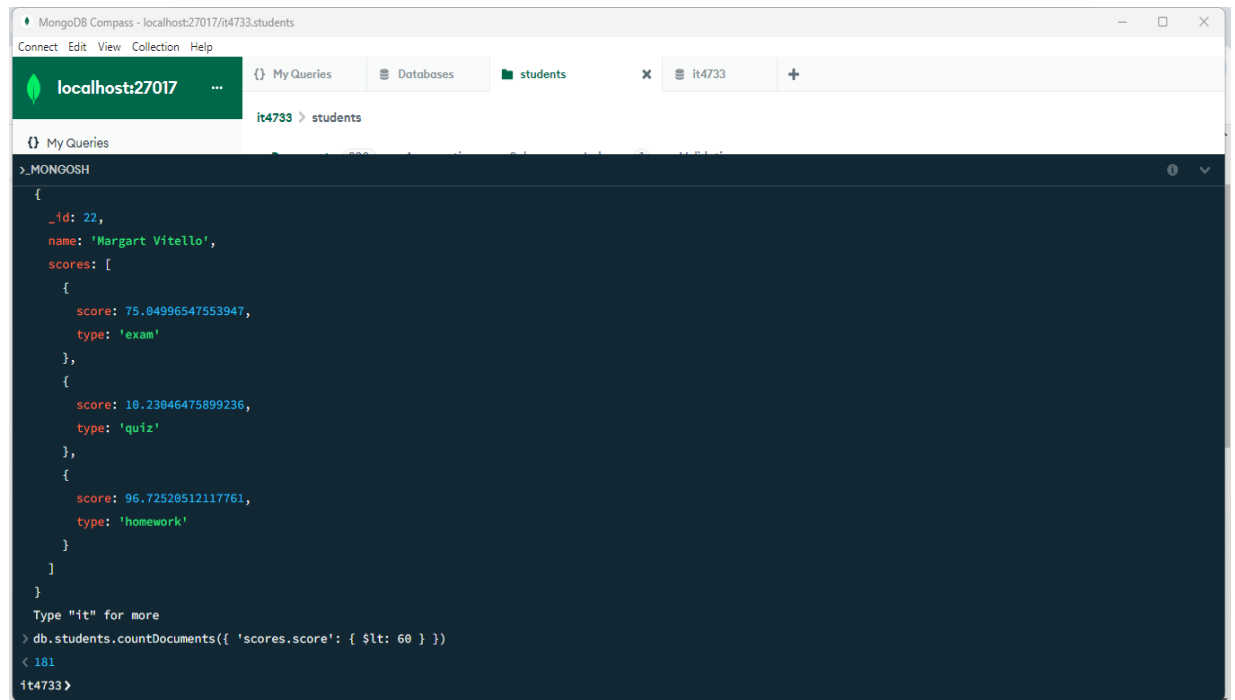
**Code:** `db.students.find({ name: { $regex: 'ski$', $options: 'i' } })`

9. Write and run a MongoDB command to find all students with at least one score below 60 (8pt)

The screenshot shows the MongoDB Compass interface. The top bar indicates the connection to 'localhost:27017' and the database 'it4733'. The 'students' collection is selected. The query editor shows the command: `db.students.find({ 'scores.score': { $lt: 60 } })`. The results pane displays two documents: one for 'amee Zank' with scores of 1.463179736705023 (exam), 11.78273389957772 (quiz), and 35.8748349954354 (homework); and another for 'Aurelia Mendez'.

**Code:** `db.students.find({ 'scores.score': { $lt: 60 } })`

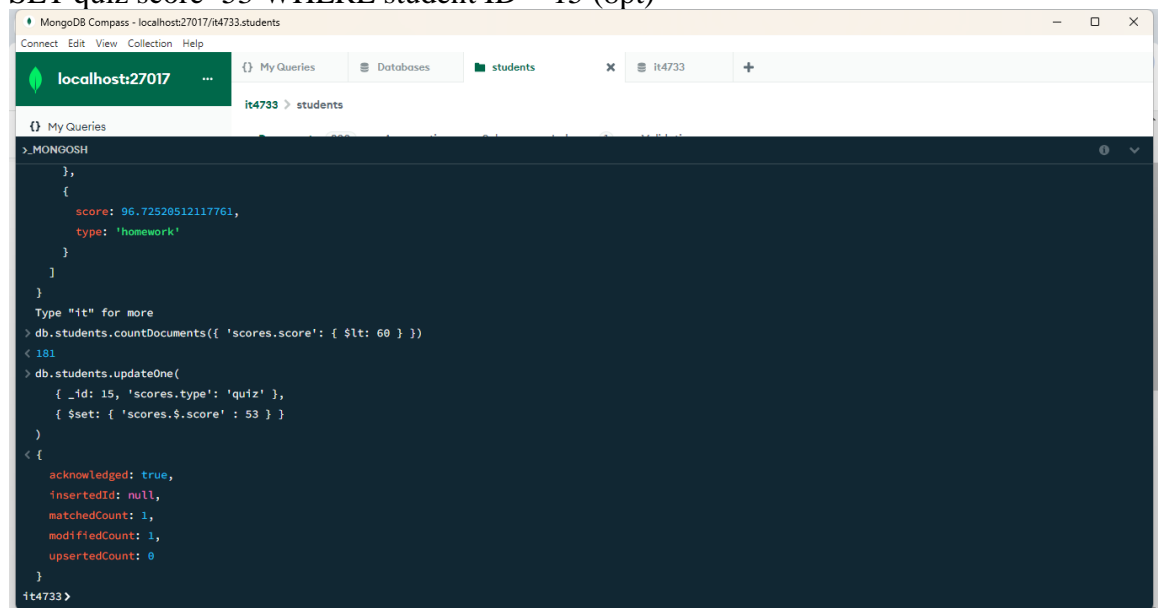
10. Write and run a MongoDB command to count all students with at least one score below 60 (8pt)



```
>_MONGOSH
{
  "_id": 22,
  "name": 'Margart Vitello',
  "scores": [
    {
      "score": 75.04996547553947,
      "type": 'exam'
    },
    {
      "score": 10.23046475899236,
      "type": 'quiz'
    },
    {
      "score": 96.72520512117761,
      "type": 'homework'
    }
  ]
}
Type "it" for more
> db.students.countDocuments({ 'scores.score': { $lt: 60 } })
< 181
tt4733>
```

**Code:** `db.students.countDocuments({ 'scores.score': { $lt: 60 } })`

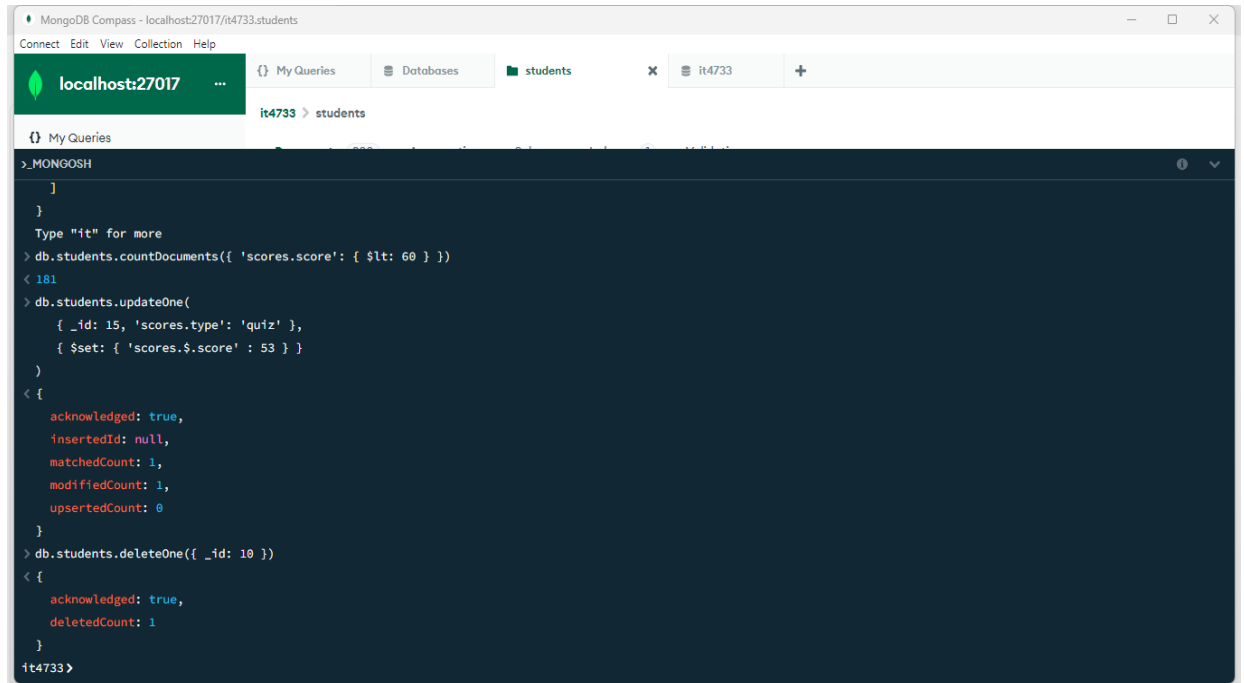
11. Write and run a MongoDB command which is equivalent to "UPDATE students SET quiz score=53 WHERE student ID = 15 (8pt)



```
>_MONGOSH
{
  "score": 96.72520512117761,
  "type": 'homework'
}
}
Type "it" for more
> db.students.countDocuments({ 'scores.score': { $lt: 60 } })
< 181
> db.students.updateOne(
  { _id: 15, 'scores.type': 'quiz' },
  { $set: { 'scores.$score' : 53 } }
)
< {
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
tt4733>
```

**Code:** `db.students.updateOne(
 { _id: 15, 'scores.type': 'quiz' },
 { $set: { 'scores.$score' : 53 } }
)`

12. Write and run a MongoDB command which is equivalent to "DELETE FROM students WHERE student id=10" (8pt)



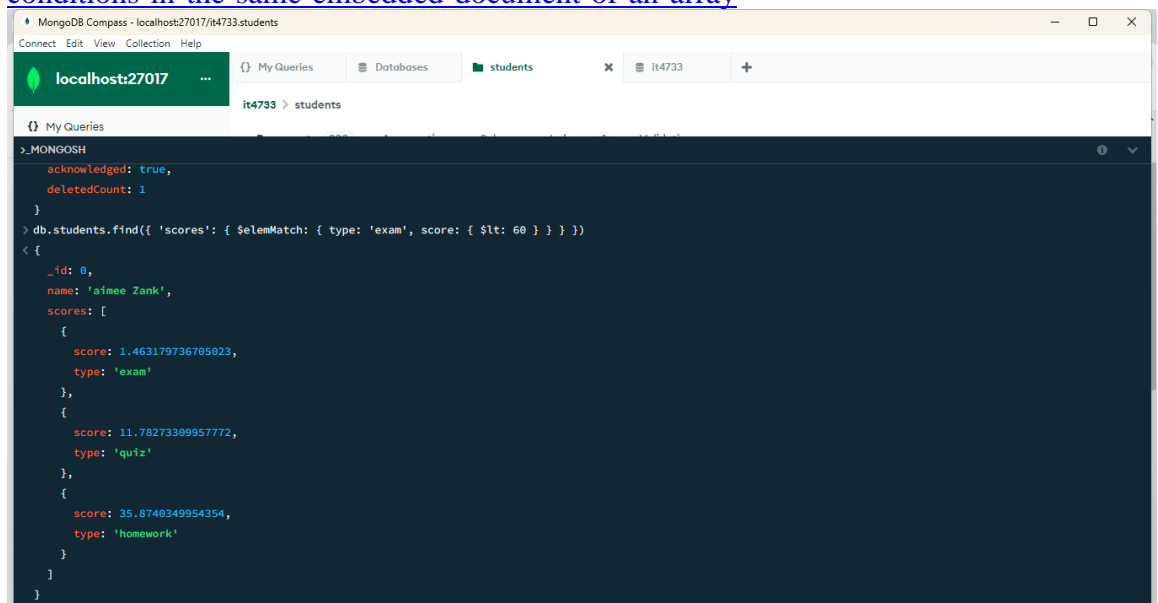
```
>_MONGOSH
]
}
Type "it" for more
> db.students.countDocuments({ 'scores.score': { $lt: 60 } })
< 181
> db.students.updateOne(
  { _id: 15, 'scores.type': 'quiz' },
  { $set: { 'scores.$.score' : 53 } }
)
< {
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
> db.students.deleteOne({ _id: 10 })
< {
  acknowledged: true,
  deletedCount: 1
}
it4733>
```

**Code:** `db.students.deleteOne({ _id: 10 })`

13. Write and run a MongoDB command which is equivalent to "SELECT \* FROM students WHERE exam score < 60" (4pt).

Hint: read the solution in this post and see if you can modify it to work

<https://stackoverflow.com/questions/39119017/query-with-and-of-two-conditions-in-the-same-embedded-document-of-an-array>

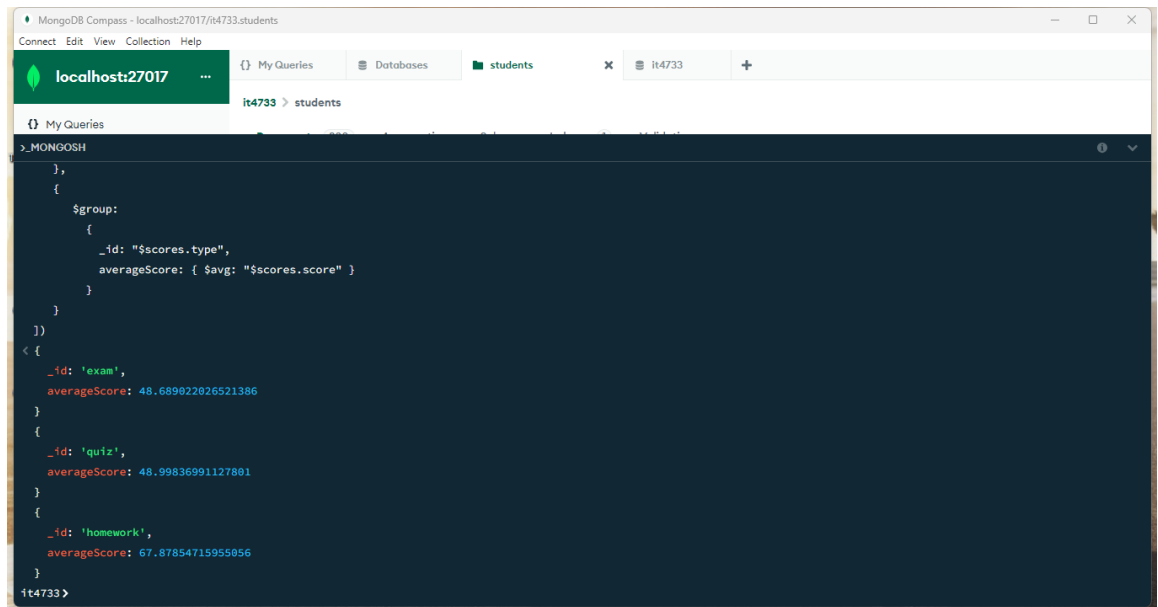


```
>_MONGOSH
acknowledged: true,
deletedCount: 1
}
> db.students.find({ 'scores': { $elemMatch: { type: 'exam', score: { $lt: 60 } } } })
< {
  _id: 0,
  name: 'aimee Zank',
  scores: [
    {
      score: 1.463179736705023,
      type: 'exam'
    },
    {
      score: 11.78273389957772,
      type: 'quiz'
    },
    {
      score: 35.8748349954354,
      type: 'homework'
    }
  ]
}
```

**Code:** `db.students.find({ 'scores': { $elemMatch: { type: 'exam', score: { $lt: 60 } } } })`

14. **Self-research question:** Write and run a MongoDB command which is equivalent to "SELECT average(exam), average(quiz), average (homework) FROM students" (4pt)





```

>_MONGOSH
{
  $group:
    {
      _id: "$scores.type",
      averageScore: { $avg: "$scores.score" }
    }
}
})
< {
  _id: 'exam',
  averageScore: 48.689822026521386
}
{
  _id: 'quiz',
  averageScore: 48.99836991127801
}
{
  _id: 'homework',
  averageScore: 67.87854715955856
}
it4733>

```

Code: db.students.aggregate([  
 {  
 \$unwind: "\$scores"  
 },  
 {  
 \$group:  
 {  
 \_id: "\$scores.type",  
 averageScore: { \$avg: "\$scores.score" }  
 }  
 }  
])